***Summary***

The testing method that I used in this project aligned with the software requirements because it tested each method that was implemented. For example, in each milestone there were aspects that had character limits or required valid future dates such as in the appointment service. To check such aspects “Assertions.assertThrows(IllegalArgumentException.class, () -> \_\_\_\_” was used. In contact Test we also tested to see if the task ID was too long, we accomplished this by using the variable “void testTaskIdTooLong”.

The quality of Junit tests is good, each week my understanding of the tests improved therefore helped the quality of my code. In my experience with this class Junit tests are very effective testing methods for a project. Junit tests have a large coverage when testing your code if implemented correctly. However Junit testing does require programming knowledge so it would be a developer that uses Junit testing as opposed to a tester. In the future I would probably practice Junit testing more in order to test my code even if I also use another testing method. Junit testing is a good method of testing your inputs in the code however you still need to be aware of other possible errors and behaviors.

I also used arrays in my code to ensure that the code was correct. Using arrays was a little easier to ensure correctness as I have had more practice using arrays. In my array list I included equals, add, and length to make the project more usable and real for the client. In the contact service I used the “Private static List <String>” array to make a list of contact IDs. I also used Junit testing to test my array values.

I often make small mistakes in my code such as missing semicolons so on top of the numerous tests I ran for the code I also checked the code a little here and there on top of checking back through the code again even after it was complete and seemed functional just to make sure all my variable names were correct and everything was in the correct place. Some of the code was tedious and therefore I made quite a few typos in my code and misplaced lines entirely. I used the resources provided in this class as well as independent research as well to verify my code to ensure I was doing it correctly. In my Junit tests I tested for correct and incorrect inputs.

***Reflection***

The testing method that I used in the milestones is a form of unit testing and white box testing. White box testing is the software testing method in which the internal structure of the program is known to the tester which includes unit testing. Unit testing is a form of testing that breaks down the code into sections and tests the sections. I achieved this by testing each area of code that accepted user input by passing an input into it and testing to see whether it would return the same input back to me.

The type of testing that I did not use for this project is non-functional testing. Non-functional testing is the testing of non-functional aspects of the project such as performance testing, security testing, or usability testing. Also, another form of testing not used is experience testing which would be based off of your own personal experience. Non-functional testing would be tested using specific tools because they cannot be tested manually.

I worked on this project with caution as I was unfamiliar with junit testing so I had to be careful when writing my code. I analyzed my code closely for an errors as I was unfamiliar with the proper include conventions. I also performed research on top of the weekly readings to effectively complete the project. In my final conclusion, Jun testing was a very effective method to use for testing the project with an over 80 percent coverage of the code.